UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, D.C. 20231
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/215,788	12/21/1998	JERRIE L. COFFMAN	21936435X00	1611

7590 06/19/2002

HUNG H. BULL
ANTONELLI, TERRY, STOUT & KRAUS
1300 N. 17TH STREET
SUITE 1800
ARLINGTON, VA 22209

EXAMINER	
PRIETO, BEATRIZ	
ART UNIT	**PAPER NUMBER**
2152	12

DATE MAILED: 06/19/2002

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/215,788	COFFMAN ET AL.	
	Examiner	Art Unit	
	B. PRIETO	2152	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133).
- Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *28 March 2002* .

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-28* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-28* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

11)☐ The proposed drawing correction filed on _____ is: a)☐ approved b)☐ disapproved by the Examiner.

If approved, corrected drawings are required in reply to this Office action.

12)☐ The oath or declaration is objected to by the Examiner.

Priority under 35 U.S.C. §§ 119 and 120

13)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____ .

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

14)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. § 119(e) (to a provisional application).

a) ☐ The translation of the foreign language provisional application has been received.

15)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. §§ 120 and/or 121.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449) Paper No(s) _____ .
4) ☒ Interview Summary (PTO-413) Paper No(s). *21* .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other:

DETAILED ACTION

1. This communication is in response to amendment filed 03/28/01, claims 1-28 remain pending.

2. Quotation of 35 U.S.C. §103(a) which forms the basis for all obviousness rejections set forth in this Office action may be found in previous office action;

3. Claims 1-18 are rejected under 35 U.S.C. 103(a) as being unpatentable over Heil et. al. (Heil) U.S. Patent No. 6,173,374 in view of Intelligent I/O (I2O) Architecture Specification (Specs), version 1.5, March 1997.

Regarding claims 1 and 14 Heil teaches a host driver configuration of a host driver module of a computer node on a server cluster, comprising:

an input/output platform (IOP) installed in a host system arranged to control an array of local/remote (I/O) storage devices;

Heil teaches a host driver module installed in a host system node (150 of Fig. 2) (col 10/line 28-50: I_20 based partitions the device driver into one module of stackable drivers 210-280 containing an operating system and hardware code for accessing/controlling an array of storage devices, host device drivers within host system node (150), col 10/lines 28-36, 51-58) of a computer node on a server cluster, (abstract: intelligent I_2O standard software, col 10/lines 51-col 11/line 4), said device driver module includes software drivers (240, 250, 260 of Fig. 2) are arranged to control an array of local storage devices, driver software (240: I/O redirector) determines whether to satisfy a block I/O request locally or remotely, and coordinates the retrieval of data over a cluster with logically shared disks, clusters drives, comprising local stored data blocks, col 11/lines 5-11; and software drivers (ISM 250/HDM 260) control local disks (118), col 11/lines 12-27, said system driver module comprising:

a software driver (230 and 240 host interfaces of Fig. 2) (Local Transport) arranged to provide an interface to said input/output platform (IOP) (col 10/lines 57-58), and software driver (240) arranged to provide an interface to said local/remote IOP, col 10/lines 66-col 11/line 35.

software drivers (I/O shipping 270 and HDM 280 of Fig. 2) (Remote Transport) arranged to provide an interface to export local storage device access onto a computer network; a host driver module architecture on a server cluster for providing I/O access storage device data transfer between computer node(s) on a server cluster system (150) and another system (151);

Heil teaches software drivers (270 and 280 of Fig. 2) is arranged to provide an interface layer to a computer network, which provides managing means to allow the host driver module (HBA) to determine whether to satisfy a block I/O request locally or remotely, managing means coordinates the retrieval of data over a cluster with logically shared disks, cluster drives, comprising local (118) or remote stored data blocks (123), col 11/lines 5-11; wherein software drivers (I/O shipping 270 and HDM 280 of Fig. 2) provide an interface layer for exporting local storage device onto a computer network, (I/O shipping 270) managing the reception of remotely generated requests from the other system network for local data to be exported onto the computer network, and (I/O shipping HDM 280) manages the communication medium (FC) transactions in support of shipping functions, col 11/lines 36-46.

software driver (I/O shipping HDM 280 of Fig. 2) (Connection Manager) arranged to establish connection services with remote servers on said computer network and coordinate functions responsible for creating a communication between the software driver (250 of Fig. 2) and software driver (240 of Fig. 2) to provide access to the local or remote storage devices .

Heil teaches establishing a communication over an FC backbone network between the initial host system on a computer node and the remote HBA host system on a computer node attached to the storage location owning the requested I/O blocks, and ships the block I/O request to the peer host driver module (HBA's) remote disk, col 11/lines 35-65, and establishes communication col 4/lines 58-61, data is transferred across computer nodes of a server cluster system by providing for I/O shipping of data blocks to peer host system, see abstract;

however Heil teachings of functions responsible for creating a communication path between the transport driver modules to provide access to the local storage devices are not

denoted a "direct call path"; nor wherein that embedded intelligent I_2O standard software comprises an "IOP platform";

Specs teaches a message-based interfaces enable direct messages passing between any two device driver module for a particular class of I/O messages class, section 1.1.2.2, page 1-4, utility message (call) functions, table page 6-4 comprising message codes for accessing devices. Upon initialization the host then gives each IOP a list of all IOPs and the physical location of their inbound message queue. When an IOP wants to connect to another IOP, it sends the request to the respective IOP's inbound message queue location. The connection request and its reply convey information enabling the two IOPs to establish a direct path for exchanging messages, section 2.1.4.1, and page 2-11.

In accordance with I_2O standard architecture, this protocol comprises a single host entity and an intelligent I/O subsystem containing a number of I/O processors entities, host comprising application(s) and their resources, executing an operating system, Fig. 2-1, an IOP platform consists of a processor, memory and I/O devices, as per architecture's standard on page 2-1.

Therefore, Heil teachings wherein an host bus adapter (HBA) executing embedded intelligent software is arranged to control an array of local storage devices, inherently teaches that an input/output platform (IOP) arranged to control an array of local storage devices.

It would have been obvious to one ordinary skilled in the art at the time the invention was made to enable means for creating a direct call path between the driver modules to provide access to the local storage device, as taught by Specs, motivation would be enable a direct message passing between software driver layer for a particular class of I/O, such as those further specified by I_20 standards such as LAN ports, Ethernet or Token ring controllers, SCSI ports, etc providing an architecture that is operating-vendor-independent and adapts to existing system, creating scalable drives from high-end workstations to high-end server, as taught by Specs.

Regarding claim 15, IOP supports at least one or more input/output processors (See Specs: I_20 standard hardware architecture is defined for a single host entity and an intelligent I/O subsystem containing a number of I/O processors entities, host comprising application(s) and their resources, executing an operating system, Fig. 2-1, an IOP platform consists of a processor, memory and I/O devices, page 2-1), and comprises:

a device driver module which interfaces the local storage devices for controlling said array of local storage devices (Heil, col 11/lines 12-35, device driver module 260); and

a communication layer which defines a mechanism for communications between the system driver module and the device driver module (see Specs: I_2O communication layer (messaging service layer), which delivers I/O transactions messages from one software module to another, any where in the I_2O domain, independent of the modules hierarchy, section 2.1.3, page 2-4, Fig. 2-3).

Regarding claim 2, IOP access module is one of a hardware module (Heil: col 11/lines 28-35), a combined hardware/software module (Heil: col 10/lines 43-50), and a software module provided on a tangible medium (Heil: col 10/lines 28-50).

Regarding claims 3 and 8, discussed on claims claim 1 and 14, further wherein a communication between host computer nodes on a server cluster system and another system (Heil. host computer nodes on a server cluster, abstract, system interconnected via a data network, col 9/lines 11-17).

Regarding claim 4, said IOP which comprises: at least one or more input/output processors (see Specs, page 2-1); at least one storage device as said input/output devices (Heil: disks 118, col 11/lines 7-35); a device driver module arranged to control access via interface means with said storage device (Heil: software driver 260, col 11/lines 7-12, 28-35); a communication layer which defines a mechanism for communications between the ISM driver 250 (Local Transport) and the HDM (260); (Heil: software driver 250, col 11/lines 12-27, supporting communication between 250 and 260, see Specs messaging (communication) layer section 2.1.3, page 2-4, Fig. 2-3).

Regarding claims 5, 6, 9, 10, and 16, wherein said communication layer is responsible for managing and dispatching all service requests (see Specs: section 2.1.3, page 2-4, Fig. 2-3 communication layer is a network of object instance) and providing a set of APIs for delivering messages along with a set of support routines that process the messages (see Specs: messaging is an interface between the modules, this interface is a set of APIs that provide transport and

message services, page 2-5), and is comprised of a message layer which sets up a communication session (see Specs: section 2.1.6, page 2.18), and transport layer which defines how information will be shared (Heil coordinate retrieval of shared information, col 10/lines 66-col 11/line 11).

Regarding claim 7, as discussed on claims 1/14, further a communication layer which defines a mechanism for communications between the system driver layer and the device driver layer (see Specs: I_2O communication layer (messaging service layer), which delivers I/O transactions messages from one software module to another, any where in the I_2O domain, independent of the modules hierarchy, section 2.1.3, page 2-4, Fig. 2-3), host system including a processor including operating system (Heil: 100 of Fig. 2, host device driver module operating system (OSM), col 10/lines 43-50 and a host system operating system col 10/lines 28-36).

Regarding claim 11, wherein said system driver module and said device driver module constitute a single device (Heil, col 10/lines 43-47, stackable device drivers in a single module) that is portable across a plurality of host operating systems and host network platforms (See Specs: to enable device drivers to port across target processors, device driver source code is written in ANSI C, section 1.2, page 1-4), and works interoperable with a plurality of storage devices and host operating systems (See Specs, coexists with existing device drivers, such as device drivers of multiple classes, and device drives can scale across system platforms, from high-end workstations to high-end server, page 1-4, C code provides portability across platforms, section 2.5.2.1, page 1-40).

Regarding claim 12, wherein said system host driver layer and said device driver layer operate in accordance with an I_20 specification for allowing storage devices to operate independently from the operating system of the host server (Heil, 10, lines 28-42, computer nodes of a server, see abstract).

Regarding claim 13, this claim is substantially the same to claim 2, as discussed, same rationale is applicable.

Regarding claim 17-18, this claims is substantially the same as claims 11-12 as discussed, same rationale is applicable.

4. Claims 19-28 are rejected under 35 U.S.C. 103(a) as being unpatentable over Heil et. al. (Heil) U.S. Patent No. 6,173,374 in view of Intelligent I/O (I_2O) Architecture Specification (Specs), version 1.5, March 1997 in further view of Bonola U.S. Patent No. 6,321,279.

Regarding claim 19, wherein, upon initialization, a software 250 driver (Local Transport) scans the local bus so as to locate and initialize all local input/output platforms (IOPs) and builds an opaque "context" structure for each input/output platform (IOP);

Specs: teach building a list of designations for each IOP, wherein after the IOP loads, it executes its initialization sequence, causing the IOP to scan its physical adapters, load and initialize appropriate devices, and build a logical configuration table, section 2.1.7.2, page 2.25, located IOP is added to the system configuration table,. host then gives each IOP a list of all IOPs, section 2.1.4.1, page 2-11, said table lists all the IOP's registered devices and their availability, item 8, page 4-65, for each device registered, the driver provides a list of message handlers, one for each message function, received messages handlers are used to builds a message dispatch table, section 2.2.4.3, page 2-32, each registered I_2O device provides message handlers that process the messages to the device, the identify of the event handler is accompanied by a context variable so that the message variable can determine the associated I/O transaction of each registered IOP, section 2.2.4.1, section Page 2-32.

wherein software driver 240 (Remote Transport) prepares to accept requests from a remote server through said computer network, (Heil: Software drivers 240 arranged to manage the reception of remotely generated requests from the network for local data to be exported onto the computer network, col 11/lines 36-46, remote computer nodes of a server cluster, abstract), and

means for building an IOP descriptor structure for each input/output platform (IOP) which includes an exported table of function call pointers and the context required by the software driver 250 to communicate with the input/output platform (IOP);

Specs: Each I_2O device is created with a unique TID and associated with an event queue, IOP and driver communicate via API function calls. Driver supplies a pointer to the (IOP descriptor function table) message dispatch table when the device is created. This table lists Function codes, their priorities and message handlers for processing request messages. When communicating with the IOP, the IOP receives a request message to that TID, the IOP uses the message's Function code (function call), queues the request to the event queue. If the driver sends requests, it must register their reply handlers (i.e. send/receive handlers) and priorities via an API function call. Handles are placed in a structure and which returns an InitiatorContext value identifying that structure. When a Driver sends a request, it uses the appropriate InitiatorContext value. When a reply is received, the IRTOS retrieves the reply handler and priority from a structure identified by the InitiatorContext field and then queues the event, section 5.1.5, page 5-4, section 5.2.6, page 5-14; In this manner an IOP descriptor structure for each input/output platform (IOP) is built, which includes an exported table of function call pointers and the context required by the driver to communicate with the input/output platform (IOP); connection using said direct call path between the driver 250 and driver 240, discussed above in claims 1 and 14;

and means for establishing a network management communication channel through the driver 240 (Heil, col 11/lines 54-65, shipping via 240, col 10/line 65-col 10/line 17), and means which waits for a connection (see Specs, table 4-5, page 4-16, connection_fail, timeout waiting for response), however neither Heil nor Specs teach determining the number of IOP;

Bonola teaches means for determining upon initialization and a starting process, the number of CPU's present in the computer system along with the context of the computer system, col 9/lines 55-58, teaching means for querying so as to determine the number of input/output processing (IOP), col 8/line 5-13, 58-64);

It would have been obvious to one ordinary skilled in the art at the time the invention was made to modify existing teachings with means for determining the number of IOP as taught by Bonola motivation would be to prevent error in the installation phase determining the actual of detected processor, utilizing said number and the context information to initialize the drivers, supporting the allocation of shared memory without requiring extra memory, as in the prior art.

Regarding claim 20, wherein said software driver 250 further has a send handler function and said driver 240 further has a receive handler function which are respective program interfaces for receiving an inbound message from a remote server on said computer network for direct access to local IOP and for delivering an outbound message to said remote server on said computer network.

Specs teaches wherein for IOP- IOP communication, inbound/outbound message queue location are exchanges, section 2.1.4.1, and page 2-11; when communicating with the IOP, the IOP receives a request message to that TID, the IOP uses the message's Function code (function call), queues the request to the event queue. If the driver sends requests, it must register their reply handlers via an API function call. When a Driver sends a request, it uses the appropriate InitiatorContext value from handles are placed in a structure and which returns an InitiatorContext value identifying that structure. When a reply is received, the reply handler are retrieved from a structure identified by the InitiatorContext field and then queues the event, section 5.1.5, page 5-4, section 5.2.6, page 5-14; In this manner an IOP descriptor structure for each input/output platform (IOP) is built, which includes an exported table of function call pointers and the context required by the driver to communicate with the input/output platform (IOP); In this manner send/receive (request/reply) handlers are exchange for corresponding inbound and outbound messages between the local and remote IOP modules.

Regarding claim 21, wherein said Remote Transport further builds an IOP connection structure including at least an IOP descriptor pointer which refers to the IOP descriptor structure of the Connection Manager for making a direct call to the Local Transport through the receive handler function and the send handler function.

Specs teaches a connection message structure is used by an IOP to connect one of its drivers and a device registered on another IOP, IOPs using send/receive request messages, messages comprising descriptor structure (e.g. InitiatorDevice and TargetDevice) used to refer to the driver and IOP that will send requests and the device on IOP2 that will receive them, section, page 4-20, said descriptor identifies the driver and the IOP, adapters and device types, section 5.2.1, page 5-7; IOP descriptor structure for each input/output platform (IOP) is built, which includes an exported table of function call (send/receive handler) pointers and the context

required by the driver to communicate with the input/output platform (IOP); connection using said direct call path between the driver 250 and driver 240, discussed above in claims 1 and 14; wherein said connection request and its reply convey information enabling the two IOPs to establish a direct path for exchanging messages, section 2.1.4.1, and page 2-11.

Regarding claim 22, as discussed on claims 1 and 14, and further

an interface to an IOP supporting an array of storage devices; (Heil: host device driver module executing embedded intelligent software is arranged to control an array of local storage devices i.e. input/output platform (IOP) arranged to control an array of local storage devices, as discussed above, col 10/line 28-col 11/line 4, said device driver module includes software drivers (240, 250, 260 of Fig. 2 are arranged to control an array of local storage devices, driver software (240: I/O redirector) coordinates the retrieval, col 11/lines 5-11, software drivers (ISM 250/HDM 260) controls local disks (118), (250) arranges to provides an interface layer to said host (I_2O standard software) driver module structure, col 11/lines 12-27, Specs, IOP supporting an array of storage devices, section 2.1.4.1, and page 2-11):

an interface to a remote server on said computer network; (Heil: software drivers (I/O shipping 270 and HDM 280 of Fig. 2) (Remote Transport) arranged to provide an interface to export local storage device access onto a computer network; a host driver module architecture on a server cluster for providing I/O access storage device data transfer between computer node(s) Fig 2, node 150 on a server cluster system (abstract) and node 151 and another system, remote a retrieval/access, col 11/lines 5-11; and I/O shipping col 11/lines 36-46, communicate with other nodes via communication medium.

establish service connection between a host server and remote server on said computer network (Fig. 5A, col 9/lines 27-48, 60-67) in response to a I/O request from a remote server on said computer network; (Heil: software driver (I/O shipping HDM 280 of Fig. 2) arranged to establish connection services with remote servers on said computer network and coordinate functions responsible for creating a communication between the software driver (250 of Fig. 2) and software driver (240 of Fig. 2) to provide access to the local or remote storage devices, locating blocks in response to requested I/O blocks (col 9/lines 1-26, I/O ship request), and ships the block I/O request to the peer host driver module (HBA's) remote disk, col 11/lines 35-

65, and establishes communication col 4/lines 58-61, data is transferred across computer nodes of a server cluster system by providing for I/O shipping of data blocks to peer host system, see abstract);

enabling said remote server to directly access said storage devices over said established service connection with transparency, bypassing operating (OS) protocol stacks installed in the host server to share resources of said storage devices (Heil: connectivity between host system (150) and (151) via established via a network communications medium (122), node-to-node inter-processor communication, col 8/lines 9-23, host device driver (181 of Fig. 5A) supports remote I/O request via connection (122), col 9/lines 27-32, 60-67, remote host device driver retrieves the requested blocks of data directly from the remote host device driver's local disk, caching these to service future local or remotely generated requests for these blocks (col 11/lines 66-col 12/lines 7), I/O shipping technique uses low-level layer of host software, i.e. is performed by the driver layer of the host software to provide physically shared resources (col 3/lines 22-32), host device driver-to-host device driver communication enable I/O shipping functionality to be removed from the host and executed by the host device driver, col 4/lines 30-40, eliminating the host system overhead while enabling host application any-to-any connectivity, I/O shipping layer within the host device driver of the clustered computer rather than in the host, where host software no longer handles to I/O shipping functions (col 5/lines 10-24), said communication is transparent to the host by the intelligent software of the host device drivers supporting remote/local I/O request, col 5/lines 53-67), communication independent of the host operating system enabling transparency to the host (col 10/lines 28-50).

Response to arguments

5. Applicant argues (A) prior art of record has no disclosure of a host driver module installed in a host system which provides an interface to an input/output platform (IOP) supporting an array of input/output devices; and arranged to interface to another system;

In response to argument (A), Heil teaches host systems and associated HBA(s) communicate in accordance with embedded (installed) intelligent device driver software, I_2O standard. (col 10/lines 28-42), I_2O standard partitions the device driver into one module

containing the required operating system specific code and a second module containing the required hardware specific code, creating stackable drivers enables insertion (installation) of intermediate service modules, (col 10/lines 43-50), for retrieving data across independent computer nodes of a server cluster by providing for I/O shipping of block level requests to peer intelligent device driver software transparent to the host independent of the operating system or hardware of the underlying network (abstract), device driver module includes software drivers (240, 250, 260 of Fig. 2) are arranged to control an array of local storage devices, driver software (240: I/O redirector) determines whether to satisfy a block I/O request locally or remotely, and coordinates (interfaces) the retrieval of data over a cluster with logically shared disks, clusters drives, comprising local stored data blocks, col 11/lines 5-11; and software drivers (ISM 250/HDM 260) control local disks (118), col 11/lines 12-27 or remote I/O request, col 9/lines 27-32, 60-37).

6. Applicant argues (B) the reasons one of ordinary skilled in the art would have been motivated to select the references and to combine them to render the claimed invention.

In response to argument B, it would have been obvious to one ordinary skilled in the art at the time the invention was made to enable means for creating a direct call path between the driver modules to provide access to the local storage device, as taught by Specs, motivation would be to support the local and remote I/O request between computer nodes in a server cluster on Heil's system, enable a direct message passing between software driver layer for a particular class of I/O, such as those further specified by I_20 standards such as LAN ports, Ethernet or Token ring controllers, SCSI ports, etc providing an architecture that is operating-vendor-independent and adapts to existing system, creating scalable drives from high-end workstations to high-end server, as taught by Specs.

7. Applicant argues (C) the prior art of record does not teach an input/output platform (IOP) module for providing input/output device access between a host system and another host system, because element (117) of the prior corresponds to applicants element NIC (328) and applicants host device driver is a separate module from the NIC.

In response to argument (C), argument that the references fail to show certain features of applicant's invention, it is noted that the features upon which applicant relies (i.e., input/output platform (IOP) module for providing input/output device access between a host system and another host system, that is a separate module from the NIC) are not recited in the rejected claim(s). Although the claims are interpreted in light of the specification, limitations from the specification are not read into the claims. See *In re Van Geuns*, 988 F.2d 1181, 26 USPQ2d 1057 (Fed. Cir. 1993).

8. Applicant argues (D) prior art does not teach because to cite two other HBA (180, 181) to correspond to the Local Transport and Remote transport components of the host device driver (IOP access module) and cite (180, 181) to correspond to connection manager component in nodes (150, 151) is improper;

In response to argument (D), Heil teaches I/O shipping software driver (HDM 280 of Fig. 2) (Connection Manager) arranged to establish connection services with remote servers on said computer network and coordinate functions responsible for creating a communication to provide access to the local or remote storage devices (col 11/lines 35-65: establishes communication in support of shipping functions, col 4/lines 58-61: data is transferred across computer nodes of a server cluster system by providing for I/O shipping of data blocks to peer host system, see abstract);

Heil teaches that node (150 of Fig. 1) includes a host system including HBA (col 6/lines 34-47, where node (150) is similar to (151), col 8/lines 50-56) that services I/O request from CPUs (100 of Fig. 1), however one skilled in the art will recognize that load sharing benefits allow multiple HBAs to service local and remote I/O request (col 7/lines 36-50), node (150 of Fig. 5A) contains two HBAs (180 and 181) directly connected via PCI bus (116.5), further connected to a node (N) via communication medium (121), (180) for processing I/O request directed to local drivers (118) and (181) to support remote I/O request (col 9/lines 27-32), HBA (181) interfaces via (193) connecting to communication medium (121) via (120). Alleged improperness is not in accordance with Heil's teachings.

9. Applicant argues (E) prior art of record does not teach a remote transport which interfaces with other nodes of said system network:

In response to argument (E), Heil teaches an interfacing to a remote server on said computer network; (Heil: software drivers (I/O shipping 270 and HDM 280 of Fig. 2) (Remote Transport) arranged to provide an interface to export local storage device access onto a computer network; a host driver module architecture on a server cluster for providing I/O access storage device data transfer between computer node(s) Fig 2, node (150) on a server cluster system (abstract) and node (151) and another system, remote a retrieval/access, col 11/lines 5-11; and I/O shipping col 11/lines 36-46, communicate with other nodes remote via communication medium.

10. Applicant argues (F) prior art does not teach an IOP descriptor structure, because analogy of prior art's directory map is incorrect.

In response to argument, claim (19) recites a "opaque context structure for each IOP...which includes an exported table of function call pointers and the context required by the Local Transport to communicate with the IOP", further according to applicant's specification to applicant's specification an context structure pointers are exchange to established a direct call relationship between software modules from a remote server, a context is a pointer to a structure in the Remote Transport which contains the address of the Remote Transport connection structure for a service connection, see page 15, lines 19-21.

Specs: teach building a list of designations for each IOP, wherein after the IOP loads, it executes its initialization sequence, causing the IOP to scan its physical adapters, load and initialize appropriate devices, and build a logical configuration table, section 2.1.7.2, page 2.25, located IOP is added to the system configuration table, host then gives each IOP a list of all IOPs, section 2.1.4.1, page 2-11, said table lists all the IOP's registered devices and their availability, item 8, page 4-65, for each device registered, the driver provides a list of message handlers, one for each message function, received messages handlers are used to builds a message dispatch table, section 2.2.4.3, page 2-32, each registered I_2O device provides message handlers that process the messages to the device, the identify of the event handler is accompanied by a context variable so that the message variable can determine the associated I/O transaction of

each registered IOP, section 2.2.4.1, section page 2-32. DMA moves blocks of data between system memory (including shared memory on other IOPs) and IOPs local memory, filling outbound frames (in the system main memory and inbound messages frames of other IOPs, utilizing API calls, see section 2.5.2.6. IOP provides the device driver module (DDM) with an API function call, the DDM receives messages addressed to that device's TID (target identifier device), and provides a message handler for each message function, the message handler is called with a pointer to the message frame, the message frame including a initiatoraddress, targetaddress, and initiator context, see section 5.1.5. See table 5-36 intiator context function including API call. DDM supplies a dispatch table to each IOP, when the I_2O device is created, table includes a list of functions codes, their priorities and message handlers for processing request messages to the IOPs, see section 5.26, page 5-14 and see Fig. 5-28, see section 5.4.6

11. Applicant argues (G) prior art of record does not teach claim limitation "service connection to a local IOP connected to a local bus using a driver module;

In response to argument (G), Heil teaches device driver module in each node, enables communication between (HBA) in the same system can communicate as peers over the system's PCI bus in accordance with the intelligent I/O standard (hereinafter referred to as the I.sub.2 O standard), col 4/lines 21-28, a single HBA or separate to support a node's local and remote storage devices (col 5/lines 58-67), software drivers (240, 250, 260 of Fig. 2) are arranged to control an array of local storage devices, driver software (240: I/O redirector) determines whether to satisfy a block I/O request locally or remotely, and coordinates the retrieval of data over a cluster with logically shared disks, col 11/lines 5-11; and software drivers (ISM 250/HDM 260) control local disks (118), col 11/lines 12-27.

12. Applicant's arguments filed 3/28/02 have been fully considered but they are not persuasive.

13. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a). A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

14. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Prieto, B. whose telephone number is (703) 305-0750. The Examiner can normally be reached on Monday-Friday from 6:00 to 3:30 p.m. If attempts to reach the examiner by telephone are unsuccessful, the Examiner's Supervisor, Mark H. Rinehart can be reached on (703) 305-4815. The fax phone number for the organization where this application or proceeding is assigned is (703) 308-6606. Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the receptionist whose telephone number is (703) 305-3800/4700. Any response to this final action should be mailed to:

Box AF
Commissioner of Patents and Trademarks
Washington, D.C. 20231

or faxed to:
(703) 746-7238, (for Official After-final communications; please mark "EXPEDITED PROCEDURE", for other Official communications; (703) 746-7239)

Or:
(703) 465-7240 (for Non-Official, Draft communications, status query, please label "PROPOSED" or "DRAFT")

Hand-delivered responses should be brought to Crystal Park II, 2121 Crystal Drive, Arlington. VA., Sixth Floor (Receptionist).

B. Prieto
Patent Examiner
June 11, 2002

LE HIEN LUU
PRIMARY EXAMINER